Accelerated Approximation of the Complex Roots and Factors of a Univariate Polynomial

Victor Y. Pan, Elias Tsigaridas

Accelerated Approximation of the Complex Roots and Factors of a Univariate Polynomial

Victor Y. Pan[1] and Elias P. Tsigaridas[2]

[1] Department of Mathematics and Computer Science
Lehman College of the City University of New York
Bronx, NY 10468 USA
and
Ph.D. Programs in Mathematics and Computer Science
The Graduate Center of the City University of New York
New York, NY 10036 USA

victor.pan@lehman.cuny.edu
http://comet.lehman.cuny.edu/vpan/

[2] INRIA, Paris-Rocquencourt Center, PolSys
Sorbonne Universités, UPMC Univ. Paris 06, PolSys,
UMR 7606, LIP6, F-75005, Paris, France
elias.tsigaridas@inria.fr

Abstract

The known algorithms approximate the roots of a complex univariate polynomial in nearly optimal arithmetic and Boolean time. They are, however, quite involved and require a high precision of computing when the degree of the input polynomial is large, which causes numerical stability problems. We observe that these difficulties do not appear at the initial stages of the algorithms, and in our present paper we extend one of these stages, analyze it, and avoid the cited problems, still achieving the solution within a nearly optimal complexity estimates, provided that some mild initial isolation of the roots of the input polynomial has been ensured. The resulting algorithms promise to be of some practical value for root-finding and can be extended to the problem of polynomial factorization, which is of interest on its own right. We conclude with outlining such an extension, which enables us to cover the cases of isolated multiple roots and root clusters.

Keywords: polynomial equation, roots, root-finding, root-refinement, power sums, complexity

1 Introduction

The classical problem of univariate polynomial root-finding has been central in Mathematics and Computational Mathematics for about four millennia since the Sumerian times, and is still important for Signal and Image Processing, Control, Geometric Modeling, Computer Algebra, and Financial Mathematics. It is closely linked to the approximation of linear and nonlinear factors of a polynomial, which is also important on its own right because of the applications to the time series analysis, Wiener

filtering, noise variance estimation, covariance matrix computation, and the study of multi-channel systems (see Wilson (1969) [35], Box and Jenkins (1976) [6], Barnett (1983) [1], Demeure and Mullis (1989 and 1990) [8], [9], and Van Dooren (1994) [34]).

Nearly optimal algorithms developed for both problems in Pan (1995) [22] and Pan (2002) [24] are quite involved, and their analysis in the cited papers ensures optimal arithmetic and Boolean complexity bounds (up to polylogarithmic factors) only if one assumes a fairly long precision of computing, of the order exceeding the degree of the input polynomial.

The most popular packages of numerical subroutines for root-finding, for a polynomial with complex coefficients, such as MPSolve 2000 (see Bini and Fiorentino (2000) [3]), EigenSolve 2001 (see Fortune (2002) [10]), and MPSolve 2012 (see Bini and Robol (2014) [5]) employ alternative root-finders based on functional iterations (namely, Ehrlich–Aberth's and WDK, that is, Weierstrass', also known as Durand-Kerner's) and the QR algorithm applied to eigen-solving for the companion matrix of the input polynomial. The user considers these root-finders practically superior by relying on the empirical data about their excellent global convergence (right from the start), even though these data have no formal support.

As in MPSolve and Eigensolve, we assume a polynomial with complex coefficients, and we re-examine this subject and show that the cited deficiency of the algorithms of [22] and [24] disappears under some mild assumptions about the initial isolation of the root sets of the input polynomial. Next we briefly comment on these results. In the next sections we elaborate upon them, analyze the algorithms, deduce the computational cost estimates, and outline some natural extensions.

Recall that polynomial root-finding iterations can be partitioned into two stages. At first a crude (although reasonably good) initial approximations to the roots or to a root are relatively slowly computed. Then these approximations are refined fast by means of the same or distinct iterations.

We partition the second stage into two substages, apply distinct efficient algorithms at these substages, and estimate their Boolean complexity. The estimates show that the computations are highly efficient and nearly optimal under the assumed mild initial isolation of a root or a root set. Empirically, such initial isolation is routinely computed by most of the known root-finders (in particular, by the popular Ehrlich–Aberth' and WDK's iterations and the QR algorithm) on their way to the approximation of the roots, although formal Boolean complexity estimates at this stage are usually missing. Thus our study raises the algorithmic challenge of most efficient computation of such a mild initial isolation.

Having reached it, the known root-finders (e.g., Ehrlich–Aberth' and WDK's iterations) typically continue the iterations as before, but at this point, we can shift to our alternative algorithm, supporting superior and nearly optimal complexity estimates. This suggests that our techniques are promising to become the user's choice.

Besides the root-finding applications, these techniques can be a valuable ingredient of the polynomial factorization algorithms. One can extend factorization to root-finding (see Schönhage (1982) [31], [22], and [24]), but Pan (2012) [25] has elaborated upon the converse reduction: namely, having sufficiently close approximations to the roots of a polynomial available, one can produce its approximate factorization into linear factors at a nearly optimal computational cost. Our present techniques can be applied into the reverse direction, in order to extend an initial mild isolation of the roots to computing the desired close approximate factorization.

Even though the root-finding algorithms of Pan [24] could be modified to obtain Boolean complexity bounds for the problem of approximating the complex roots up to any desired precision, to our knowledge, there are no (nearly) optimal Boolean complexity results for dedicated algorithms for approximating (all) the complex roots of univariate polynomials. Our results (Theorem 8 and 10) could be seen as an extension of the recently obtained bounds for approximating the real roots [28, 29], see also [17].

An interesting research challenge is the design of a polynomial factorization algorithms that both are simple enough for practical implementation and support factorization at a nearly optimal computational complexity. A promising approach to this goal, outlined in our last section, is the combination of our present algorithms with the one of Pan 2012 [25] and McNamee and Pan 2013 [18, Section 15.23], which is a simplified version of the efficient, but very much involved algorithm

of Kirrinnis (1998) [13].

A preliminary version of this paper has been presented at SNC 2014.

Organization of the paper. We recall the relevant definitions and some basic results in the remainder of this section and in the next section. In Section 3 we present our main algorithm, prove its correctness, and estimate its arithmetic cost when it is aplied to the approximation of a single root and d simple isolated roots of a dth degree polynomial. In Section 4 we extend our analysis to estimate the Boolean cost of these computations. In our concluding Section 5 we summarize our results and then outline their extension to factorization of a polynomial and to root-finding in the cases of isolated multiple roots and root clusters.

Some definitions.

- For a polynomial $u = u(x) = \sum_{i=0}^{d} u_i x^i$, the norms $||u||_\gamma$ denote the norms $||\mathbf{u}||_\gamma$ of its coefficient vector $\mathbf{u} = (u_i)_{i=0}^{d}$, for $\gamma = 1, 2, \infty$.
- $D(X, r)$ denotes the complex disc $\{x : |x - X| \le r\}$.
- "ops" stands for "arithmetic operations".
- $DFT(q)$ denotes the discrete Fourier transform at q points. It can be performed by using $O(q \log(q))$ ops.

2 Isolation Ratio and Root-refinement

The following concept of the *isolation ratio* is basic for us, as well as for [22] and [24]. Assume a real or complex polynomial

$$p = p(x) = \sum_{i=0}^{d} p_i x^i = p_n \prod_{j=1}^{d} (x - z_j), \quad p_d \neq 0, \tag{1}$$

of degree d, an annulus $A(X, R, r) = \{x : r \le |x - X| \le R\}$ on the complex plane with a center X, and the radii r and R of the boundary circles. Then the internal disc $D(X, r)$ is *R/r-isolated*, and we call R/r its *isolation ratio* if the polynomial p has no roots in the annulus. The isolation ratios, for all discs $D(0, r)$ and for all positive r, can be approximated as long as we can approximate the root radii $|z_j|$, for $j = 1, \dots, d$.

The algorithms of [31] (cf. also Pan (2000) [23] and [24]) yield such approximations within a constant relative error, say, 0.01, by using $O(d \log^2(d))$ ops, but involve the Dandelin's root-squaring iteration (see Householder (1959) [12]), and this leads to numerical stability problems.

Alternative heuristic algorithms of Bini (1996) [2] and [3] are slightly faster, but also cannot produce close approximation without using root-squaring iteration.

The Schur-Cohn test does not use these iterations and can be applied to estimate the isolation ratio more directly. For the disc $D(0, r)$, a variant of this test in Renegar (1987) [30, Section 7] amounts to performing FFT at $d' = 2^h$ points, for $16d \le d' \le 32d$, with the overhead of $O(n)$ ops and comparisons of real numbers with 0. This means a reasonably low precision of computing and Boolean cost. Also see Brunie and Picart (2000) [7].

The next result from Tilli (1998) [32] shows that Newton's classical iteration converges quadratically to a single simple root of p if it is initiated at the center of a $3d$-isolated disc that contains just this root. The result softens the restriction that $s \ge 5d^2$ of [30, Corollary 4.5].

Theorem 1. *Suppose that both discs $D(c, r)$ and $D(c, r/s)$, for $s \ge 3d$, contain a single simple root α of a polynomial $p = p(x)$ of degree d. Then Newton's iteration*

$$x_{k+1} = x_k - p(x_k)/p'(x_k), k = 0, 1, \dots \tag{2}$$

converges quadratically to the root α right from the start provided that $x_0 = c$.

3 Increasing Crude Isolation Ratios of Polynomial Roots

We can shift and scale the variable x, and so with no loss of generality we assume dealing with a $(1+\eta)^2$-isolated disc $D(0,r)$, for $r = 1/(1+\eta)$, a fixed $\eta > 0$, and a polynomial p of (1) having precisely k not necessarily distinct roots $z_1, \dots, z_k$ in this disc.

In Section 5 we outline some important extensions of our current study based on our results in the general case of any $k < d$, which we produce next, but in Sections 3 and 4 we assume that $k = 1$.

Now, under the above assumptions for any $k < d$, can we increase the isolation ratio, say, to $3d$? We apply the following algorithm.

Algorithm 2. The Power Sums.

INPUT: three integers d, k and q, such that $0 < k < n$, $k < q$, and $\eta > 0$,

$\omega = \omega_q = \exp(2\pi\sqrt{-1}/q)$, a primitive qth root of unity,

the coefficients $p_0, \dots, p_d$ of a polynomial $p = p(x)$ of (1).

We write $r = 1/(1+\eta)$ and assume that the disc $D(0,r)$

(i) is $(1+\eta)^2$-isolated and

(ii) contains the k roots $z_1, \dots, z_k$ of the polynomial $p = p(x)$ and no other roots.

OUTPUT: the values $\sigma_1^*, \dots, \sigma_k^*$ such that

$$|\sigma_g - \sigma_g^*| \le \Delta_{k,q} = (r^{q+k} + (d-1)r^{q-k})/(1-r^q), \text{ for } g = 1, \dots, k, \tag{3}$$

$$\sigma_g = \sum_{j=1}^{k} z_j^g, \text{ for } g = 1, \dots, k. \tag{4}$$

COMPUTATIONS:

1. Compute the coefficients of the two auxiliary polynomials $p_q(x) = \sum_{i=0}^{l} p_{q,i} x^i$ and $\bar{p}_q(x) = \sum_{i=0}^{l} \bar{p}_{q,i} x^i$ where $p_{q,i} = \sum_{j=0}^{l} p_{i+jq}$ and $\bar{p}_{q,i} = \sum_{j=0}^{\bar{l}} p_{i+1+jq}$, for $i = 0, \dots, q-1$, $l = \lfloor d/q \rfloor$ and $\bar{l} = \lfloor (d-1)/q \rfloor$.
2. Compute the values $p_q(\omega^j)$, $\bar{p}_q(\omega^j)$, and $r_j = p_q(\omega^j)/\bar{p}_q(\omega^j)$, for $j = 0, 1, \dots, q-1$,
3. Compute and output the values $\sigma_g^* = \frac{1}{q} \sum_{j=0}^{q-1} \omega^{(g+1)j} r_j$, for $g = 1, \dots, k$.

Theorem 3. *Algorithm 2 involves $3d + O(q \log(q))$ ops.*

Proof. Stage 1 of the algorithm involves $d-1$ multiplications and less than $2d$ additions, Stage 2 amounts to performing two DFT(q) and q divisions, and Stage 3 amounts to performing an DFT(q) (because ω^{g+1}, for $g = 1, \dots, k$, is the set of all qth roots of 1) and k divisions. □

In order to prove bound (3), implying *correctness of the algorithm*, at first observe that $p(\omega^j) = p_q(\omega^j)$ and $p'(\omega^j) = \bar{p}_q(\omega^j)$, for $j = 0, 1, \dots, q-1$, and hence

$$\sigma_g^* = \frac{1}{q} \sum_{j=0}^{q-1} \omega^{(k+1)j} p(\omega^j)/p'(\omega^j), \text{ for } g = 1, \dots, k. \tag{5}$$

The proof of bound (3) also exploits the Laurent expansion

$$p'(x)/p(x) = \sum_{j=1}^{d} \frac{1}{x - z_j} = -\sum_{g=1}^{\infty} \sigma'_g x^{g-1} + \sum_{g=0}^{\infty} \sigma_g x^{-g-1} = \sum_{h=-\infty}^{\infty} c_h x^h \tag{6}$$

where $|x| = 1$, $\sigma_0 = 1$, $\sigma_g = \sum_{j=1}^{k} z_j^g$ (cf. (4)), $\sigma'_g = \sum_{i=k+1}^{d} z_i^{-g}$, $g = 1, 2, \dots$, that is, σ'_g is the gth power sum of the roots of the reverse polynomial $p_{\mathrm{rev}}(x)$ that lie in the disc $D(0,r)$. The leftmost

equation of (6) is verified by the differentiation of $p(x) = p_n \prod_{j=1}^{d}(x - z_j)$. The middle equation is implied by the decompositions $\frac{1}{x-z_1} = \frac{1}{x}\sum_{h=0}^{\infty}\left(\frac{z_1}{x}\right)^h$ and $\frac{1}{x-z_i} = -\frac{1}{z_i}\sum_{h=0}^{\infty}\left(\frac{x}{z_i}\right)^h$, for $i > 1$, provided that $|x| = 1$ for all i. (Note a link of these expressions with the following quadrature formulae for numerical integration, $\sigma_g = \frac{1}{2\pi\sqrt{-1}}\int_{C(0,1)} x^m p'(x)/p(x)dx$, for $g = 1, \dots, k$.)

In order to deduce bound (3), we next combine equations (5) and (6) and obtain

$$\sigma_k^* = \sum_{l=-\infty}^{+\infty} c_{-k-1+lq}.$$

Moreover, equation (6), for $h = -k-1$ and $k \geq 1$, implies that $\sigma_k = c_{-k-1}$, while the same equation, for $h = k-1$ and $k \geq 1$, implies that $\sigma_k' = -c_{k-1}$. Consequently

$$\sigma_k^* - \sigma_k = \sum_{l=1}^{\infty}(c_{lq-k-1} + c_{-lq-k-1}).$$

We assumed in (5) that $0 < k < q-1$. It follows that $c_{-lq-k-1} = \sigma_{lq+k}$ and $c_{lq-k-1} = -\sigma_{lq-k}'$, for $l = 1, 2, \dots$, and we obtain

$$\sigma_k^* - \sigma_k = \sum_{l=1}^{\infty}(\sigma_{lq+k} - \sigma_{lq-k}'). \tag{7}$$

Now recall that $|\sigma_h| \leq z^h$ and $|\sigma_h'| \leq (d-1)z^h$, for $h = 1, 2, \dots$ and $z = \max_{j=1}^{d}\min(|z_j|, 1/|z_j|)$, and so $z \leq \frac{1}{1+t}$ in our case. Substitute these bounds into (7) and obtain

$$|\sigma_k^* - \sigma_k| \leq (z^{q+k} + (d-1)z^{q-k})/(1-z^q).$$

Therefore, bound (3) follows because $z \leq r$.

By substituting q of order $\log(d)$ into bound (3), we can increase the isolation ratio of the disc $D(0,r)$ by a factor of gd^h, for any pair of positive constants g and h. Therefore we obtain the following estimates.

Theorem 4. *Suppose the disc $D(0,r) = \{x : |x| \leq r\}$ is $(1+\eta)^2$-isolated, for $(1+\eta)r = 1$ and a fixed $\eta > 0$, and contains exactly k roots of a polynomial $p = p(x)$ of degree d. Let g and h be a pair of positive constants. Then it is sufficient to perform $O(3d + \log(d)\log(\log(d)))$ ops in order to compute a gd^h-isolated subdisc of $D(0,r)$ containing exactly the same roots of $p = p(x)$.*

If $k = 1$, then $\sigma_1 = z_1$, and by combining Theorems 1 and 4 we obtain the following result.

Corollary 5. *Let a polynomial $p(x)$ satisfy the assumptions of Theorem 4, for $k = 1$. Then we can approximate its root within ϵ, for $0 < \epsilon < 1$, by using $O(\log(d)\log(\log(d)) + d\log(\log(1/\epsilon)))$ ops.*

Hereafter we write

$$z_+ = \max(|z_1|, |z_2|, \dots, |z_d|). \tag{8}$$

Corollary 6. *Suppose that we are given d discs, each containing a single simple root of a polynomial $p = p(x)$ of degree d and each being $(1+\eta)^2$-isolated, for a fixed $\eta > 0$. Then we can approximate all d roots of this polynomial within ϵz_+, for z_+ of (8) and a fixed ϵ, $0 < \epsilon < 1$, by using $O(d\log^2(d)(1 + \log(\log(1/\epsilon))))$ ops.*

Proof. Apply Algorithm 2 concurrently in all d given discs, but instead of the qth roots of unity use q equally spaced points at the boundary circle of each input disc (that is, $dq = O(d\log d)$ points overall) and instead of DFT(q) apply the Moenck–Borodin algorithm for multipoint polynomial evaluation [20].

Also use it at the stage of performing concurrent Newton's iteration initialized at the centers of the $3d$-isolated subdiscs of the d input discs, each subdisc computed by the algorithm that supports Theorem 4. Here we work with the dth degree polynomial p rather than with the qth degree

polynomials p_q. Indeed, in order to support transition to polynomials p_q of degree q, for d discs, we would need to perform d shifts and scalings of the variable x, which would involve the order of $d^2 \log(d)$ ops, whereas by employing the Moenck–Borodin algorithm, we obtain a nearly optimal root-refiner, which involves $O(d)$ ops up to polylogarithmic factors.

Technically, we replace the matrix $\Omega = [\omega^{j(k+1)}]_{j,k}$ in (5) by the matrix $[c+\omega^{j(k+1)}]_{j,k} = c[1]_{j,k}+\Omega$ where c is invariant in j and k. The multiplication of the new matrix by a vector $\mathbf{v}$ is still reduced to multiplication of the matrix Ω by a vector $\mathbf{v}$ with the additional $3d$ ops, for computing the vector $c[1]_{j,k}\mathbf{v}$ and adding it to the vector $\Omega\mathbf{v}$. □

The Moenck–Borodin algorithm uses nearly linear arithmetic time, and [13] proved that this algorithm supports multipoint polynomial evaluation at a low Boolean cost as well (see also J. van der Hoeven (2008) [33], Pan and Tsigaridas (2013a,b) [28], [29], Kobel and Sagraloff (2013) [14], Pan (2015) [26], and Pan (2015a) [27]). Consequently *our algorithm supporting Corollary 6 can be extended to support a nearly optimal Boolean cost bound for refining all simple isolated roots of a polynomial*.

Instead of Newton's, one can apply various other iterative root-refiners (see McNamee (2002) [15], McNamee (2007) [16], and McNamee and Pan (2013) [18]). They also support our complexity estimates as long as our power sum algorithm yields isolation of the roots sufficient in order to ensure superlinear global convergence of the selected iterations. In particular Ehrlich–Aberth's and WDK iterations converge globally with cubic and quadratic rate, respectively, if all the d discs have isolation ratios at least $3\sqrt{d}$, for Ehrlich–Aberth's iterations, and $8d/3$, for WDK iterations (cf. [32]).

Remark 7. The algorithm of [26] and [27] (the latter paper provides more details) approximates a polynomial of degree d at $O(d)$ points within ϵz_+, for z_+ of (8) and fixed ϵ such that $0 < \epsilon < 1$, by using $O(d \log^2(d)(1 + \log(1/\epsilon)))$ ops. This matches the bound of [20], for $1/\epsilon$ of order gd^h and for positive constants g and h. Moreover the algorithm of [26] and [27], performed with the IEEE standard double precision, routinely outputs close approximations to the values of the polynomial $p(x)$ of degree $d = 4096$, at d selected points, whereas the algorithm of [20] routinely fails numerically, for d of about 40 (cf. [27]).

4 Boolean Cost Bounds

Hereafter $\widetilde{O}_B$ denotes the bit (Boolean) complexity ignoring logarithmic factors. By $\lg(\cdot)$ we denote the logarithm with base 2. To estimate the Boolean complexity of the algorithms supported by Corollaries 5 and 6 we apply some results from Pan and Tsigaridas (2013) [28] and Pan and Tsigaridas (2015) [29], which hold in the general case where the coefficients of the polynomials are known up to an arbitrary precision. In this section we assume that the polynomial $p = p(x)$ has Gaussian (that is, complex integer) coefficients known exactly; the parameter λ, to be specified in the sequel, should be considered as the working precision. We assume that we perform the computations using fixed point arithmetic.

At first we consider the algorithm of approximating one complex root, z, of a polynomial p up to any desired precision ℓ. By an approximation we mean absolute approximation, that is compute a $\tilde{z}$ such that $|z - \tilde{z}| \le 2^{-\ell}$. We assume that the degree of p is d and that $\|p\|_\infty \le 2^\tau$.

Following the discussion that preceded Theorem 4, we compute the polynomial p_q and then apply two DFTs, for p_q and $\bar{p}_q$, and the inverse DFT, for $p_q/\bar{p}_q$.

Assume that p is given by its λ-approximation $\widetilde{p}$ such that $\lg\|p - \widetilde{p}\|_\infty \le -\lambda$. Perform all the operations with $\widetilde{p}$ and keep track of the precision loss to estimate the precision of computations required in order to obtain the desired approximation.

We compute p_q using d additions. This results in a polynomial such that

$$\lg\|p_q\|_\infty \le \tau + \lg(d)$$

and

$$\lg(\|p_q - \widetilde{p}_q\|_\infty) \le -\lambda + \tau \lg(d) + 1/2 \lg^2(d) + 1/2 \lg(d) = O(-\lambda + \tau \log(d) + \log^2(d)).$$

Similar bounds hold for p'_q, that is,

$$\lg(\|p'_q\|_\infty) \le \tau + 2\lg(d)$$

and

$$\lg(\|p'_q - \widetilde{p}'_q\|_\infty) \le -\lambda + \tau\lg(d) + 3/2\lg^2(d) + 1/2\lg(d) = O(-\lambda + \tau\log(d) + \log^2(d)).$$

The application of DFT on p'_q leads us to the following bounds,

$$|p'_q(\omega^i)| \le \tau + 2\lg(d) + \lg\lg(d) + 2 = O(\tau + \log(d))$$

and

$$|p'_q(\omega^i) - \widetilde{p'_q(\omega^i)}| \le -\lambda + \tau\lg(2d) + 3/2\lg^2(d) + 5/2\lg(d) + \lg\lg(d) + 5 = O(-\lambda + \tau\log(d) + \log^2(d)),$$

for all i, [29, Lemma 16]. Similar bounds hold for $p_q(\omega^i)$.

The divisions $k_i = p_q(\omega^i)/p'_q(\omega^i)$ output complex numbers such that

$$|k_i| = |p_q(\omega^i)/p'_q(\omega^i)| \le \tau + 2\lg d + \lg\lg d + 2.$$

Define their approximations $\tilde{k}_i$ such that

$$\lg(|k_i - \widetilde{k}_i|) \le -\lambda + \tau\lg(4d) + 3/2\lg^2 d + 9/2\lg d + 2\lg\lg d + 11 = O(-\lambda + \tau\log(d) + \log^2(d)).$$

The final DFT produces numbers such that the logarithms of their magnitudes are not greater than $\tau + 2\lg d + 2\lg\lg d + 4$ and the logarithms of their approximation errors are at most $-\lambda + \tau\lg(8d) + 3/2\lg^2 d + 13/2\lg d + 4\lg\lg d + 18 = O(-\lambda + \tau\log(d) + \log^2(d))$, [29, Lemma 16].

To achieve an error within $2^{-\ell}$ in the final result, we perform all the computations with accuracy $\lambda = \ell + \tau\lg(8d) + 3/2\lg^2 d + 13/2\lg d + 4\lg\lg d + 18$, that is $\lambda = O(\ell + \tau\log d + \log^2 d) = \widetilde{O}(\ell + \tau)$.

We perform d additions at the cost $O_B(d\,\lambda)$ and perform the rest of computations, that is the 3 DFTs, at the cost $O_B(\log(d)\,\log\log(d)\,\mu(\lambda))$ or $\widetilde{O}_B(d(\ell + \tau))$ [29, Lemma 16]. If the root that we want to refine is not in the unit disc, then we replace τ in our bounds with $d\tau$.

We apply a similar analysis from [28, Section 2.3] to the Newton iteration (see also [29, Section 2.3]) and arrive at the same asymptotic bounds on the Boolean complexity.

In [28] and [29] the error bounds of Newton operator have been estimated by using the properties of real interval arithmetic. In this paper we perform our computation in the field of complex numbers, but this affects only the constants of interval arithmetic, and so asymptotically, both the error bounds and the complexity bounds of the Newton iterations are the same. Thus, the overall complexity is $\widetilde{O}_B(d^2\tau + d\ell)$ and the working precision is $O(d\tau + \ell)$.

In our case we also assume the exact input, that is, assume the coefficients of the input polynomials known up to arbitrary precision; for example, they are integers. For the refinement of the root up to the precision of L bits, we arrive at an algorithm that supports the following complexity estimates.

Theorem 8. *Under the assumptions of Theorem 4 we can approximate the root z of the polynomial $p(x) \in \mathbb{Z}[x]$, which is of degree d and $\|p\|_\infty \le 2^\tau$, up to precision of L bits in $\widetilde{O}_B(d^2\tau + dL)$.*

If we are interested in refining all complex roots, we cannot work anymore with the polynomial p_q of degree $q = O(\lg d)$ unless we add the cost of d shifts of the initial approximations to the origin. Instead we rely on fast algorithms for multipoint evaluation. Initially we evaluate the polynomial p of degree d at $O(d\lg d)$ points, and we assume that $\lg\|p\|_\infty \le \tau$. These d points approximate the roots of p, and so their magnitude is at most $\le 2^\tau$.

We use the following result of [29, Lemma 21]. Similar bounds appear in [13, 14, 33].

Theorem 9 (Modular representation)**.** *Assume that we are given $m+1$ polynomials, $F \in \mathbb{C}[x]$ of degree $2mn$ and $P_j \in \mathbb{C}[x]$ of degree n, for $j = 1, \dots, m$ such that $\|F\|_\infty \le 2^{\tau_1}$ and all roots of the polynomials P_j for all j have magnitude of at most 2^ρ. Furthermore assume λ-approximations of F by $\widetilde{F}$ and of P_j by $\widetilde{P}_j$ such that $\|F - \widetilde{F}\|_\infty \le 2^{-\lambda}$ and $\|P_j - \widetilde{P}_j\|_\infty \le 2^{-\lambda}$. Let $\ell = \lambda - O(\tau_1 \lg m + m\,n\,\rho)$. Then we can compute an ℓ-approximations $\widetilde{F}_j$ of $F_j = F \mod P_j$, for $j = 1, \dots, m$, such that $\|F_j - \widetilde{F}_j\|_\infty \le 2^{-\ell}$ in $\widetilde{O}_B(m\,n\,(\ell + \tau_1 + m\,n\,\rho))$.*

Using this theorem we bound the overall complexity of multipoint evaluation by $\widetilde{O}_B(d(L + d\tau))$. The same bound holds at the stage where we perform Newton's iteration. We need to apply Newton's operator $\widetilde{O}(1)$ for each root. Each application of the operators consists of two polynomial evaluations. We perform the evaluations simultaneously and apply Theorem 9 to bound the complexity. On similar estimates for the refinement of the real roots see [29].

We have the following theorem, which complements Corollary 6 with the Boolean complexity estimates.

Theorem 10. *Suppose that we are given d discs, each containing a single simple root of a polynomial $p(x) \in \mathbb{Z}[x]$ of degree d and $\|p\|_\infty \le 2^\tau$, and each being $(1+\eta)^2$-isolated, for a fixed $\eta > 0$. Then we can approximate all d roots of this polynomial within L bits in in $\widetilde{O}_B(d^2\tau + dL)$.*

5 Conclusions

We have begun our study of polynomial root-finding with partitioning the algorithms into stages.

(i) At first, one computes some crude initial approximations to the roots of a polynomial of a degree d by applying some algorithms, such as Ehrlich–Aberth's and WDK iterations, which have excellent empirical record of fast and reliable global convergence, even though it is not supported by formal analysis so far.

(ii) Then one obtains some initial isolation of the roots and root clusters, by using the crude approximations computed at stage (i). This stage requires further study, although some initial progress was reported in [25].

(iii) Next one increases the isolation in order to ensure fast convergence of the algorithms applied at the final stage.

(iv) Finally, one applies Newton's, Ehrlich–Aberth's, or WDK iterations or a simplified version of the algorithm of [13] presented in [25] and [18, Section 15.23].

Our current contribution is at stage (iii). Namely, we measure the isolation of a root or a root cluster by the isolation ratio of the covering disc, that is, by the ratio of the distance from the disc center to the external roots and the disc radius. Then our algorithm (performed at the nearly optimal arithmetic and Boolean cost) increases the isolation ratio from any constant $1 + \eta$, for a positive η, to gd^h, for any pair of positive constants g and h. If such an isolation is ensured, then Newton's, Ehrlich–Aberth's, and WDK iterations of stage (iv) converge right from the start with quadratic or cubic rate (cf. [32]).

Our analysis can be readily extended to prove the same results even if we assume weaker initial isolation with the ratio as low as $1 + \eta$, for η of order $1/\log(n)$. This enables transition to stages (ii) and (iii) after fewer iterations of stage (i). Further extension of our results to the case of η of order $1/n^h$, for a positive constant h, is a theoretical challenge, but progress at stage (i) and is more important for the theory, while progress at stages (ii) and (iv) is more important for both theory and practice. In particular we are looking for the extension of stage (iv) to computing polynomial factorization, which is important on its own right (see the first paragraph of the introduction) and would cover root-finding in the cases of multiple and clustered roots. Let us conclude with some comments on this subject.

Our Theorem 4 covers the case where we are given a $(1 + \eta)$-isolated disc containing k roots of a polynomial $p(x)$ of (1). In that case, at the cost of performing $O(d + \log(d)\log(\log(d)))$ ops, Algorithm 2 outputs approximations to the power sums of these roots within the error bound $\Delta_{q,k}$ of order r^{q-k}, for $r = 1/(1+\eta)$ and positive integers $q = O(d)$ and $k < q$ of our choice, say, $q = 2k$. The algorithm can be extended to the case where all roots or root clusters of the polynomial $p(x)$

are covered by s such $(1+\eta)$-isolated discs. Then we can still approximate the power sums of the roots in all discs simultaneously by using $O(d\log^2(d))$ ops.

At this point the known algorithm for the transition from the power sums to the coefficients (cf. [4, Problem I·POWER·SUMS, pages 34–35]) approximates the coefficients of the s factors of $p(x)$ whose root sets are precisely the roots of $p(x)$ lying in these s discs. The algorithm performs within dominated arithmetic and Boolean cost bounds.

Having these factors computed, one can reduce root-finding for $p(x)$ to root-finding for the s factors of smaller degree. In particular, this would cover root-finding in the harder case where the discs contain multiple roots or root clusters. As this has been estimated in [17] and [25], such a divide and conquer approach can dramatically accelerate stage (iv), even versus the Ehrich–Aberth's and WDK algorithms, except that the initialization of the respective algorithm requires a little closer approximation to the coefficients of the s factors of $p(x)$ than we obtain from the power sums of their roots.

The algorithm of [13], extending the one of [31], produces such a desired refinement of the output of Algorithm 2 at a sufficiently low asymptotic Boolean cost, but that algorithm of [13] is quite involved. In particular, it reduces all operations with polynomials to multiplication of long integers. In our further work we will seek the same asymptotic complexity results at this stage, but with smaller overhead constants, based on the simplified version of the algorithm presented in [25] and [18, Section 15.23].

Acknowledgments. Both authors thank the reviewers for their comments that improved the presentation of the paper. VP has been supported by NSF Grant CCF 1116736 and by PSC CUNY Award 67699-00 45. ET has been partially supported by GeoLMI (ANR 2011 BS03 011 06), HPAC (ANR ANR-11-BS02-013) and an FP7 Marie Curie Career Integration Grant.

References

[1] S. Barnett, *Polynomial and Linear Control Systems*, Marcel Dekker, New York, 1983.

[2] D. Bini, Numerical Computation of Polynomial Zeros by Means of Aberth's Method, *Numerical Algorithms* **13**, 179–200, 1996.

[3] D. A. Bini, G. Fiorentino, Design, Analysis, and Implementation of a Multiprecision Polynomial Rootfinder, *Numerical Algorithms*, **23**, 127–173, 2000.

[4] D. Bini, V. Y. Pan, *Polynomial and Matrix Computations*, Volume 1: *Fundamental Algorithms* (XVI + 415 pages), Birkhäuser, Boston, 1994.

[5] D.A. Bini, L. Robol, Solving Secular and Polynomial Equations: A Multiprecision Algorithm *J. Computational and Applied Mathematics*, **272**, 276–292, 2014.

[6] G. E. P. Box, G. M. Jenkins, *Time Series Analysis: Forecasting and Control*, Holden-Day, San Francisco, 1976.

[7] C. Brunie, P. Picart, A Fast Version of the Schur–Cohn Algorithm, *Journal of Complexity*, **16**, **1**, 54–69, 2000.

[8] C. J. Demeure, C. T. Mullis, The Euclid Algorithm and the Fast Computation of Cross–covariance and Autocovariance Sequences, *IEEE Trans. Acoust., Speech, Signal Processing* **37**, 545–552, 1989.

[9] C. J. Demeure, C. T. Mullis, A Newton–Raphson Method for Moving-average Spectral Factorization Using the Euclid Algorithm, *IEEE Trans. Acoust., Speech, Signal Processing* **38**, 1697–1709, 1990.

[10] S. Fortune, An Iterated Eigenvalue Algorithm for Approximating Roots of Univariate Polynomials, *J. of Symbolic Computation*, **33**, **5**, 627–646, 2002.

[11] M. Fürer, Faster Integer Multiplication. *SIAM J. on Computing*, **39, 3**, 979–1005, 2009.

[12] A. S. Householder, Dandelin, Lobachevskii, or Graeffe, *American Mathematical Monthly* **66**, 464–466, 1959.

[13] P. Kirrinnis, Polynomial Factorization and Partial Fraction Decomposition by Simultaneous Newton's Iteration, *J. of Complexity* **14**, 378–444 (1998).

[14] A. Kobel and M. Sagraloff, Fast Approximate Polynomial Multipoint Evaluation and Applications, arXiv:1304.8069v1 [cs.NA] 30 April 2013.

[15] J.M. McNamee, A 2002 Update of the Supplementary Bibliography on Roots of Polynomials, *J. of Computational and Applied Math.* **142**, 433-434, 2002; also at web-site `www.yorku.ca/~mcnamee/`

[16] J.M. McNamee, *Numerical Methods for Roots of Polynomials (Part 1)*, Elsevier, Amsterdam, 2007.

[17] J. M. McNamee, V.Y. Pan, Efficient Polynomial Root-refiners: A Survey and New Record Efficiency Estimate, *Computers and Mathematics with Applications*, **63**, 239–254, 2012.

[18] J. M. McNamee, V.Y. Pan, *Numerical Methods for Roots of Polynomials*, Part 2 (XXII + 718 pages), Elsevier, Amsterdam, 2013.

[19] K. Mehlhorn, M., Sagraloff, P. Wang, From Approximate Factorization to Root Isolation with Application to Cylindrical Algebraic Decomposition, in *Proc. International Symp. on Symbolic and Algebraic Computations, (ISSAC 2013)*, Boston, Massachusetts, June 2013, (M. Kauers, editor), 283-290, ACM Press, New York (2013).

[20] R. Moenck, A. Borodin, Fast Modular Transform via Division, *Proc. 13th Ann. Symp. Switching Automata Theory*, 90–96, IEEE Comp. Soc. Press, Washington, DC, 1972.

[21] V.Y. Pan, *How to Multiply Matrices Faster. Lecture Notes in Computer Science*, **179**, Springer, Berlin, 1984.

[22] V. Y. Pan, Optimal (up to Polylog Factors) Sequential and Parallel Algorithms for Approximating Complex Polynomial Zeros, *Proc. 27th Ann. ACM Symp. on Theory of Computing (STOC '95)*, ACM Press, New York, 741–750 (1995).

[23] V. Y. Pan, Approximating Complex Polynomial Zeros: Modified Quadtree (Weyl's) Construction and Improved Newton's Iteration, *J. of Complexity* **16, 1**, 213–264, 2000.

[24] V. Y. Pan, Univariate Polynomials: Nearly Optimal Algorithms for Factorization and Rootfinding, *Journal of Symbolic Computations*, **33**, **5**, 701–733, 2002.

[25] V. Y. Pan, Root-refining for a Polynomial Equation, Proceedings of *CASC 2012* (V. P. Gerdt, V. Koepf, E. W. Mayr, and E. V. Vorozhtsov, editors), *Lecture Notes in Computer Science*, **7442**, 271–282, Springer, Heidelberg (2012).

[26] V. Y. Pan, Transformations of Matrix Structures Work Again, *Linear Algebra and Its Applications*, **465**, 1–32, 2015.

[27] V. Y. Pan, Fast Approximate Computations with Cauchy Matrices and Polynomials, arXiv:1506.02285 [math.NA] (32 pages, 6 figures, 8 tables), 7 June 2015. Proc. versions in Proceedings of *CASC 2013* (V. P. Gerdt, V. Koepf, E. W. Mayr, and E. V. Vorozhtsov, editors), Lecture Notes in Computer Science, **8136**, 273–287, Springer, Heidelberg (2013), and in Proceedings of *CSR 2014* (E.A. Hirsch et al., editors), Lecture Notes in Computer Science **8476**, 287–300, Springer International Publishing, Switzerland 2014).

[28] V. Y. Pan, E. P. Tsigaridas, On the Boolean Complexity of the Real Root Refinement, in *Proc. International Symp. on Symbolic and Algebraic Computations, (ISSAC 2013)*, Boston, Massachusetts, June 2013, (M. Kauers, editor), 299–306, ACM Press, New York (2013).

[29] V. Y. Pan, E. P. Tsigaridas, Nearly Optimal Refinement of Real Roots of a Univariate Polynomial, *J. of Symbolic Computation*, http://dx.doi.org/10.1016/j.jsc.2015.06.009 (2015).

[30] J. Renegar, On the Worst-case Arithmetic Complexity of Approximating Zeros of Polynomials, *J. of Complexity* **3, 2**, 90–113 (1987).

[31] A. Schönhage, The Fundamental Theorem of Algebra in Terms of Computational Complexity, Preliminary Report, Mathematisches Institut der Universität Tübingen, Germany, 1982.

[32] P. Tilli, Convergence Conditions of Some Methods for the Simultaneous Computations of Polynomial Zeros, *Calcolo*, **35**, 3–15, 1998.

[33] J. van der Hoeven, Fast Composition of Numeric Power Series, Tech. Report 2008-09, Université Paris-Sud, Orsay, France, 2008.

[34] P. M. Van Dooren, Some Numerical Challenges in Control Theory, *Linear Algebra for Control Theory, IMA Vol. Math. Appl.* **62**, 1994.

[35] G. T. Wilson, Factorization of the Covariance Generating Function of a Pure Moving-average Process, *SIAM J. on Numerical Analysis* **6**, 1–7, 1969.